AD-A060 199   NAVY UNDERWATER SOUND LAB NEW LONDON CONN   F/G 9/2
A GENERAL PURPOSE MULTIDIMENSIONAL FAST FOURIER TRANSFORM FOR U--ETC(U)
SEP 68 R L GORDON, N L OWSLEY

UNCLASSIFIED   USL-TM-2242-319-68   NL

1 OF 1
AD
A060 199

END
DATE
FILMED
12-78
DDC

(NavShips) J. D. Pope

MOST Project -3

Code No. 00V1C

Copy No. 71

AD A060199

DDC FILE COPY

USL LIBRARY COPY

2242-319-68

001198

U. S. NAVY UNDERWATER SOUND LABORATORY
FORT TRUMBULL, NEW LONDON, CONNECTICUT

A GENERAL PURPOSE MULTIDIMENSIONAL FAST FOURIER TRANSFORM FOR USE IN FORTRAN V PROGRAMS*

by

R. L. Gordon and N. L. Owsley

USL Technical Memorandum No. 2242-319-68

6 September 1968

DDC
OCT 23 1978
F

LEVEL II

INTRODUCTION

Fast Fourier Transform (FFT)

An external subroutine is available which performs the discrete Fourier Transform on a multidimensional array of floating point data. The data may be either real or complex. The transform values are always complex and are returned to the array used to carry the original data. The dimensionality of the input array is limited only by machine storage capacity. The number of input samples is not limited to a power of two and, in fact, can be prime in any or all dimensions. The subroutine is called by a one line, six argument call statement in the user program.

Procedure for FFT

The FFT subroutine is called with the following statement:

```
CALL FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)
```

*The Program described in this memo has been adapted for the UNIVAC 1108 by the authors from a Lincoln Laboratory Technical Note "Three Programs That Perform the Cooley-Tukey Fourier Transform" by N. M. Brenner on 28 July 1967. The AD No. is AD 657019.

This document has been approved for public release and sale; its distribution is unlimited.

254200

where:

DATA = An N-dimensional complex array used to hold the real and imaginary parts of the input data, and the real and imaginary parts of the transformed values. The real and imaginary parts of a datum must be stored in adjacent core locations as done by the complex array declarer in Fortran V. For real input data, the array DATA is a complex array with the imaginary part set to 0.0.

NN = A single dimensional integer array equal in length to the number of dimensions in the transform. The first element in the array is equal to the length of the first data dimension, the second element is equal to the length of the second data dimension, etc.

NDIM = The number of dimensions used for the array DATA.

ISIGN = $\begin{cases} -1 \text{ for a forward transform.} \\ +1 \text{ for an inverse transform.} \end{cases}$

IFORM = + 1 if data and transform are complex.
0 is data is real (zero imaginary values).
If it is "0" the imaginary parts of the data should be set to zero. (Transform will be complex).

WORK = "0" if all dimensions are equal to a power of two. If not it must be a single dimension floating point array equal to twice the length of the largest data array dimension not a power of two.

Special Notes:

1. There are no error messages, error halts or error returns in this subroutine. If NDIM or any NN(I) is less than one, the program returns immediately.

2. The data is assumed to form one cycle of a periodic function.[2]

3. The data is assumed to be equispaced in each dimension.[4] If the equispaced interval is DT, the resulting transform will be equispaced from 0 to $2\pi(N-1)/(NDT)$. The upper limit is identified with $-2\pi/(NDT)$ and all points above the foldover frequency π/NDT are identified with the corresponding negative frequency.

ACCESSION for
NTIS White Section
DDC Buff Section
UNANNOUNCED
JUSTIFICATION for the on file
DISTRIBUTION/AVAILABILITY CODES
Dist. AVAIL. and/or SPECIAL
A

4. If an inverse transform is performed on an array of transformed data, the original data will reappear multiplied by N1*N2* . . .

5. The running time is extremely fast if the dimensions are rich in factors of two. The particular running time is dependent on the factorization of each dimension. Approximate times for a one dimensional complex transform are as follows:

N	Factorization	Time for Complex Transform (SEC)
4094	$2 \times 23 \times 89$	40
4096	$3^2 \times 5 \times 7 \times 13$	12
4096	2^{12}	3.1
4097	17×241	90
4098	$2 \times 3 \times 683$	240
4099	PRIME	1430
4100	$2^2 \times 5^2 \times 41$	20

6. For the use of real data input the designation IFORM = 0 can reduce running time by as much as forty percent.

7. A user must allow about 1750 core locations for the instructions.

8. The source and relocatable elements are on File 2 of Cur Tape U-183.

Examples of FFT Program

A. One dimensional forward transform of real data of length 64.

```
    COMPLEX D (64)                    @ Declare Data Array
    Do 10 I = 1, 64
 10 D(I) = (R, 0.0)                   @ Zero Imaginary Values

    - - - - - - - - -  )
    - - - - - - - - -  |
    - - - - - - - - -  }   Fortran Program
    - - - - - - - - -  |
    - - - - - - - - -  )

    CALL FOURT (D, 64, 1, -1, 0, 0)
```

where D = Data Array
64 = Length of Array
1 ~ One-Dimension
1 ~ Forward Transform
0 ~ Data is real
0 ~ No working array needed.

B. Three-dimensional forward Fourier transform of a complex array dimensioned 32 by 25 by 13.

```
DIMENSION WORK (50), NN (3)
COMPLEX D (32, 25, 13)          @ Declares Data  Array
DATA NN/32, 25, 13/             @ Supplies Dimension Info to NN

- - - - - - - - -  )
- - - - - - - - -  )
- - - - - - - - -  )  Fortran Program
- - - - - - - - -  )
- - - - - - - - -  )

CALL FOURT (D, NN, 3, -1, 1, WORK)
```

where:

D = Data Array
NN = Integer Array of Length 3 to Supply Dimension Info
3 ~ Three Dimensional Transform
-1 ~ Forward Transform
1 ~ Complex Transform
WORK = Array of size 50 (2 x 25) since 25 is largest non power of two dimension.

Conclusions

The discrete Fourier Transform presented is extremely flexible in nature. Because of this flexibility considerable storage is required compared to some special purpose FFT's. It is felt that the main use of this FFT should be as an analytical tool as opposed to a "real time" processing technique.

R. L. Gordon
R. L. GORDON

N. L. Owsley
N. L. OWSLEY

References

1. Brenner, N. M., "Three Fortran Programs That Perform the Cooley-Tukey Transform", Lincoln Laboratory Tech Note 1967-2, 28 July 1967, AD-657019.

2. Marsh, H. W., "Fast Fourier Transform - Theory and Application", Notes, April 3, 1967.

3. Haavind, R., "The FFT Computer: Designers 'Missing Link'", Electronic Design, Dec. 6, 1967.

4. Cooley, J. W. and Tukey, J. W., "An Algorithm for the Machine Calculation of Complex Fourier Series", Math. of Comput. XIX, April 1965, pp. 297-301.

DISTRIBUTION LIST
3ND-USNUSL-142

DATE	FILE	SERIAL
2 October 1968	9-A-415-00-00/175	2242-319-68

SUBJECT
A GENERAL PURPOSE MULTIDIMENSIONAL FAST FOURIER TRANSFORM FOR USE IN FORTRAN V PROGRAMS by R. L. Gordon and N. L. Owsley

TECHNICAL MEMORANDUM 2242-319-68	EVALUATION REPORT
CALIBRATION MEMORANDUM	SERVICE TEST REPORT
CALIBRATION REPORT	TEST REPORT
LETTER	OTHER
CLASSIFICATION UNCLASSIFIED	NUMBER OF COPIES MADE 71

COPY NO.		COPY NO.	
1	A. J. Alfiero, Code 2072	17	A. Nuttall, Code 2020
2	R. J. Almeida, Code 2072	18	R. M. Forget, Code 2072
3	C. R. Arnold, Code 2070	19	M. J. Goldstein, Code 2072
4	J. W. Askintowicz, Code 2072	20	S. Horvitz, Code 2072
5	J. A. Auwood	21	R. C. Jennings
6	M. D. Becker, Code 2072	22	J. F. Kelly, Code 2072
7	P. A. Breslin, Code 2072	23	J. R. Kelly (UNIVAC)
8	R. A. Brewer	24	R. J. Lambird, Code 2005
9	W. E. Cable, Code 2071	25	A. A. Lesick
10	D. L. Card	26	H. E. Lundy, Code 2072
11	R. D. Clark, Code 2072	27	L. P. Onyx, Code 2072
12	W. R. Cleary (UNIVAC)	28	E. M. Podeszwa, Code 2040
13	J. S. Cohen, Code 2072	29	E. F. Poole (UNIVAC)
14	D. W. Counsellor, Code 2072	30	D. T. Porter, Code 2221
15	R. D. Drinkard, Code 2072	31	M. R. Powers, Code 2072
16	L. T. Einstein, Code 2071	32	D. L. Rawson, Code 2072

REMARKS

COPY NO.		COPY NO.	
33	F. E. Robbins, Code 2072	59	R. L. Gordon, Code 2242
34	D. Hyde, Code 2020	60	R. L. Gordon, Code 2242
35	N. Owsley, Code 2242	61	Code 2064.2
36	N. Owsley, Code 2242	62	Code 2064.2
37	N. Owsley, Code 2242	63	Code 2064.2
38	G. R. Swope, Code 2072	64	Code 2064.2
39	M. A. Tucchio, Code 2333	65	Code 2064.2
40	T. T. Tylaska, Code 2072	66	Code 2064.2
41	A. R. Umphrey	67	Code 2064.2
42	J. F. Underwood	68	Code 2064.2
43	H. Weinberg, Code 2071	69	Code 2064.2
44	T. Wheeler, Code 2072	70	Code 2064.2
45	R. D. Whittaker, Code 2072	71	NavShips, Code OOV1C - J. D. Pope
46	E. Windt, Code 2072	72	
47	H. S. Newman, Code 2242	73	
48	R. Kneipfer, Code 2242	74	
49	J. Ianniello, Code 2213	75	
50	A. Quazi, Code 2242	76	
51	T. McAndrews, Code 2241	77	
52	A. Krajec, Code 2241	78	
53	F. J. Kingsbury, Code 2242	79	
54	Code 2242-S	80	
55	Code 2242-S	81	
56	R. L. Gordon, Code 2242	82	
57	R. L. Gordon, Code 2242	83	
58	R. L. Gordon, Code 2242	84	

REMARKS